Tutoraufgabe 1 (Longest Common Subsequence):

Bestimmen Sie die *längste gemeinsame Teilsequenz* der Sequenzen `AACHEN` und `ANDERNACH`. Benutzen Sie hierfür den in der Vorlesung vorgestellten Algorithmus mit dynamischer Programmierung und füllen Sie die folgende Tabelle aus.

Lösung:

Die Tabelle wird vom Algorithmus wie folgt gefüllt:

	∅	A	N	D	E	R	N	A	C	H
∅	0	0	0	0	0	0	0	0	0	0
A	0	↖ 1	← 1	← 1	← 1	← 1	← 1	1	1	1
A	0	1	1	1	1	1	1	↖ 2	2	2
C	0	1	1	1	1	1	1	2	↖ 3	3
H	0	1	1	1	1	1	1	2	3	↖ 4
E	0	1	1	1	2	2	2	2	3	↑ 4
N	0	1	2	2	2	2	3	3	3	↑ 4

Also erhalten wir die Sequenz `AACH` als längste gemeinsame Teilsequenz der Sequenzen `AACHEN` und `ANDERNACH`.

Tutoraufgabe 2 (Dynamische Programmierung):

a) Schreiben Sie eine Funktion, welche die Länge der Longest Common Subsequence (LCS) für zwei gegebene Zeichensequenzen berechnet. Die Zeichensequenzen seien dabei der Einfachheit halber `int` Arrays `seq1` und `seq2` mit den Längen `l1` and `l2`. Demnach soll Ihre Funktion die folgende Signatur haben:

```
int lcs(int seq1[], int l1, int seq2[], int l2)
```

b) Betrachten Sie folgendes Szenario. Ihnen wird eine Klausur gestellt, welche n Aufgaben umfasst. Jede dieser Aufgaben hat eine Punktzahl p_i und eine von Ihnen geschätzte Zeit t_i, die Sie zum Lösen der Aufgabe benötigen ($1 \leq i \leq n$). Geben Sie die maximale Punktzahl, welche Sie in T Zeiteinheiten erreichen können, als Rekursionsgleichung an (inklusive der passenden Argumente).

Lösung:

a)
```
int lcs(int seq1[], int l1, int seq2[], int l2) {
    int L[l1+1,l2+1];
    for (int i = 0; i <= l1; i++) {
        L[i,0] = 0;
    }
    for (int j = 0; j <= l2; j++) {
        L[0,j] = 0;
    }
    for (int i = 1; i <= l1; i++) {
        for (int j = 1; j <= l2; j++) {
            if (seq1[i] == seq2[j]) {
```

```
                L[i,j] = L[i - 1, j - 1] + 1;
            } else if (L[i - 1, j] > L[i, j - 1]) {
                L[i,j] = L[i - 1, j];
            } else {
                L[i,j] = L[i, j - 1];
            }
        }
    }
    return L[l1,l2];
}
```

b) Wir definieren die Funktion P in Abhängigkeit von der Anzahl i an betrachteten Aufgaben und der zur Verfügung stehenden Zeit T.

$$P(i,T) = \begin{cases} 0 & \text{falls } i = 0 \\ max(P(i-1,T), P(i-1,T-t_i) + p_i) & \text{falls } T \geq t_i \\ P(i-1,T) & \text{sonst} \end{cases}$$

Die maximal erreichbare Punktzahl ist damit $P(n,T)$.

Tutoraufgabe 3 (Graham-Scan):

Berechnen Sie die konvexe Hülle der folgenden Punktmenge. Benutzen Sie dafür *Grahams Scan* wie *in der Vorlesung* vorgestellt und geben Sie die Teilschritte *nach jeder Iteration* (also nach jedem neu hinzugefügten Punkt) an. Umkreisen Sie die Punkte, die vom Algorithmus in der Iterationsschleife nicht betrachtet werden.

Lösung: